(22F) CST8277 Enterprise Application Programming

Assignment 1: DataBank-JSF-JDBC

Please read this document carefully. If your submission does not meet the requirements as stated here, you may lose marks even though your program runs. Additionally, this assignment is also a teaching opportunity, that is **materials presented in this document may be on the midterm and/or the final exam!**
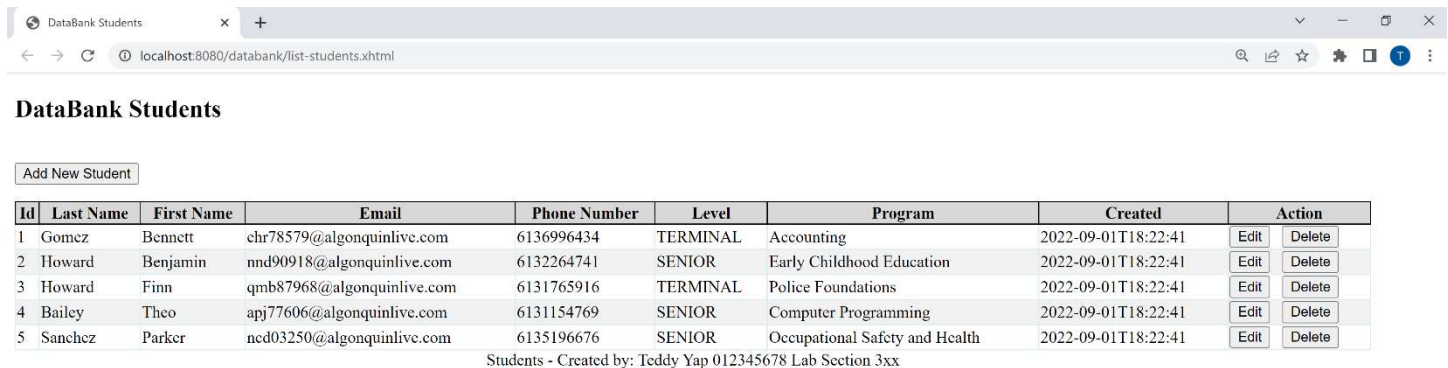
Assignment 1's submission is to be uploaded to the Assignment 1 Brightspace submission folder. It does not matter if it is a zip file containing a zip file containing... many-levels-of-nested-zip-files or if you upload the required artifacts individually, Brightspace will handle things. If you have a 100% perfect solution <u>but it is not in the correct folder,</u> you will receive a mark of zero ☹ If you know that you made mistake, please upload it to the correct folder ASAP.

Overview

For Assignment 1, assume you are hired by ACME Corp. as a programmer. The company wants you to finish developing their Student Directory web application where users should be able to:

1. **C**reate/add a new student
2. **R**ead/view a list of students
3. **U**pdate/edit an existing student
4. **D**elete/remove an existing student

The main page of the web application is shown below. The main page shows the list of students already on the database.

Id	Last Name	First Name	Email	Phone Number	Level	Program	Created	Action	
1	Gomez	Bennett	chr78579@algonquinlive.com	6136996434	TERMINAL	Accounting	2022-09-01T18:22:41	Edit	Delete
2	Howard	Benjamin	nnd90918@algonquinlive.com	6132264741	SENIOR	Early Childhood Education	2022-09-01T18:22:41	Edit	Delete
3	Howard	Finn	qmb87968@algonquinlive.com	6131765916	TERMINAL	Police Foundations	2022-09-01T18:22:41	Edit	Delete
4	Bailey	Theo	apj77606@algonquinlive.com	6131154769	SENIOR	Computer Programming	2022-09-01T18:22:41	Edit	Delete
5	Sanchez	Parker	ncd03250@algonquinlive.com	6135196676	SENIOR	Occupational Safety and Health	2022-09-01T18:22:41	Edit	Delete

Students - Created by: Teddy Yap 012345678 Lab Section 3xx

Clicking the "Add New Student" button on the main page should redirect the user to the Add New Student page shown below:

Add New Student

Last Name [   ]
First Name [   ]
Email [   ]
Phone Number [   ]
Level [FRESHMAN   ∨]
Program [Accounting   ∨]
[Add New Student] Back

Clicking the "Add New Student" button (after entering all the necessary information), should insert the new student to the database and redirect the user back to the main page. Clicking the "Back" button above should simply cancel the add operation and redirect the user back to the main page.

Clicking the "Edit" button on the main page should redirect the user to the "Update Student" page shown below:

Update Student

Last Name [Sanchez   ]
First Name [Parker   ]
Email [ncd03250@algonquinlive.cc]
Phone Number [6135196676   ]
Level [SENIOR   ∨]
Program [Occupational Safety and Health   ∨]
[Edit] Back

Clicking the "Edit" button (after updating all the necessary information), should update the existing student on the database and redirect the user back to the main page. Clicking the "Back" button above should simply cancel the update operation and redirect the user back to the main page.

Clicking the "Delete" button on the main page should trigger a pop-up to confirm if the user wants to proceed with deleting the selected student. Clicking OK on the pop-up should remove the selected student from the database as well as from the list of students shown on the main page. Clicking the "Cancel" button should simply cancel the delete operation.

DataBank Students

Add New Student

Id	Last Name	First Name	Email	Phone Number	Level	Program	Created	Action	
1	Gomez	Bennett	ehr78579@algonquinlive.com	6136996434	TERMINAL	Accounting	2022-09-01T18:22:41	Edit	Delete
2	Howard	Benjamin	nnd90918@algonquinlive.com	6132264741	SENIOR	Early Childhood Education	2022-09-01T18:22:41	Edit	Delete
3	Howard	Finn	qmb87968@algonquinlive.com	6131765916	TERMINAL	Police Foundations	2022-09-01T18:22:41	Edit	Delete
4	Bailey	Theo	apj77606@algonquinlive.com	6131154769	SENIOR	Computer Programming	2022-09-01T18:22:41	Edit	Delete
5	Sanchez	Parker	ncd03250@algonquinlive.com	6135196676	SENIOR	Occupational Safety and Health	2022-09-01T18:22:41	Edit	Delete

Students - Created by: Teddy Yap 012345678 Lab Section 3xx

Finally, the company wants you to display your student information on the main page (below the table) so that they would know who to contact if something goes wrong while using the web application.

Theme for Assignment 1

After completing this assignment, you will have achieved the following:

1. Create a model class to represent a database row
2. Create a managed bean class to manipulate the model class
3. Create a DAO class to manipulate the database
4. Create a JSF application that supports **C**reate – **R**ead – **U**pdate – **D**elete lifecycle for instances of the model class

Starting Assignment 1

ACME Corp. has provided you with a starter code *DataBank-JSF-JDBC-Skeleton.zip* to complete. You must use the file *DataBank-JSF-JDBC-Skeleton.zip* from Brightspace to start your solution. You need to extract its contents to some work folder on your hard drive.

The starter code/skeleton project is an Eclipse Maven project. There is a screencast on Brightspace that shows how to import an existing Maven project into your Eclipse workspace (Note: It is from an earlier version of the course but it is still relevant). From the *File* menu, choose *Import…*
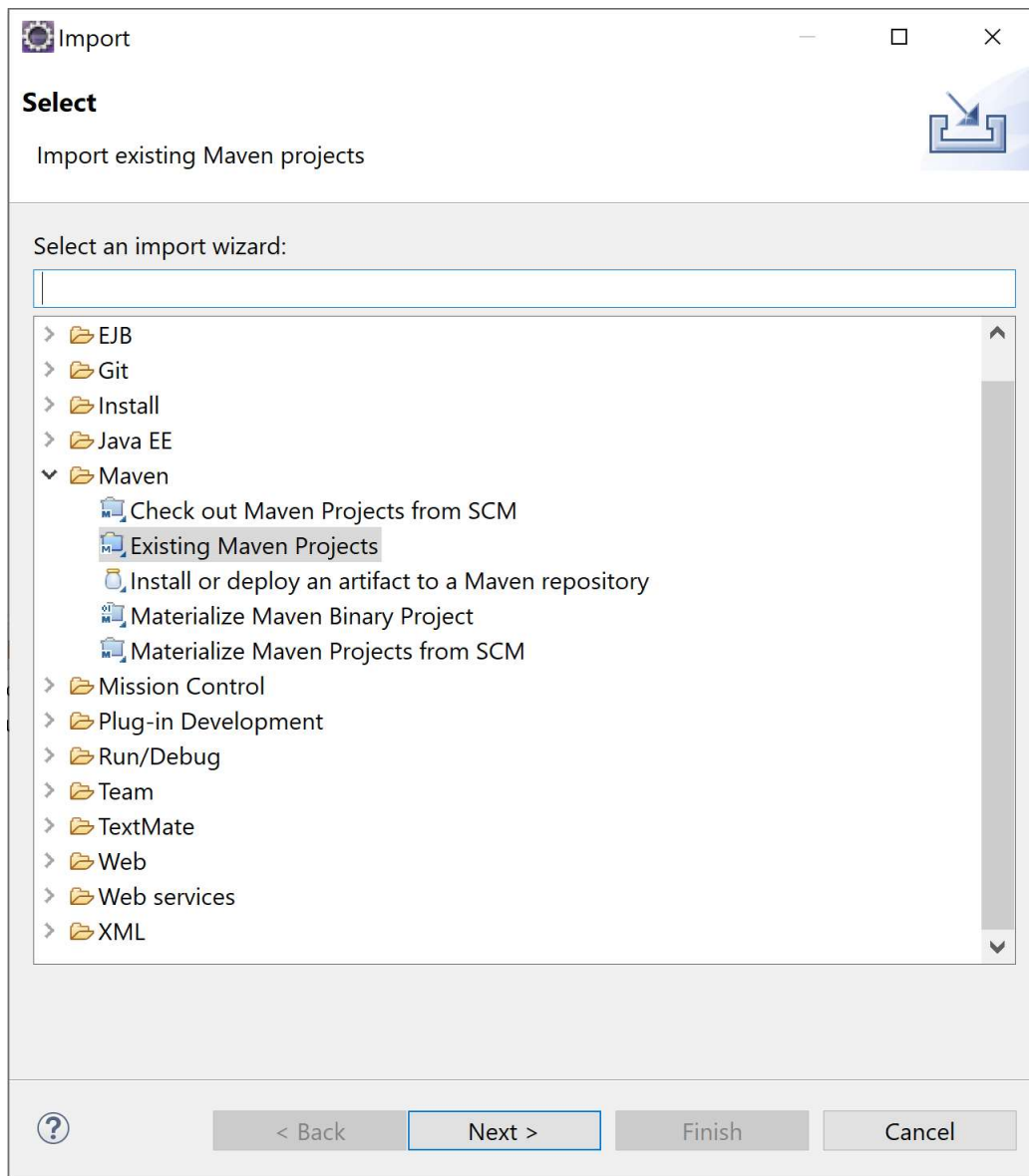
Make sure to choose the "Existing Maven Projects" on the import wizard above and click "Next >".
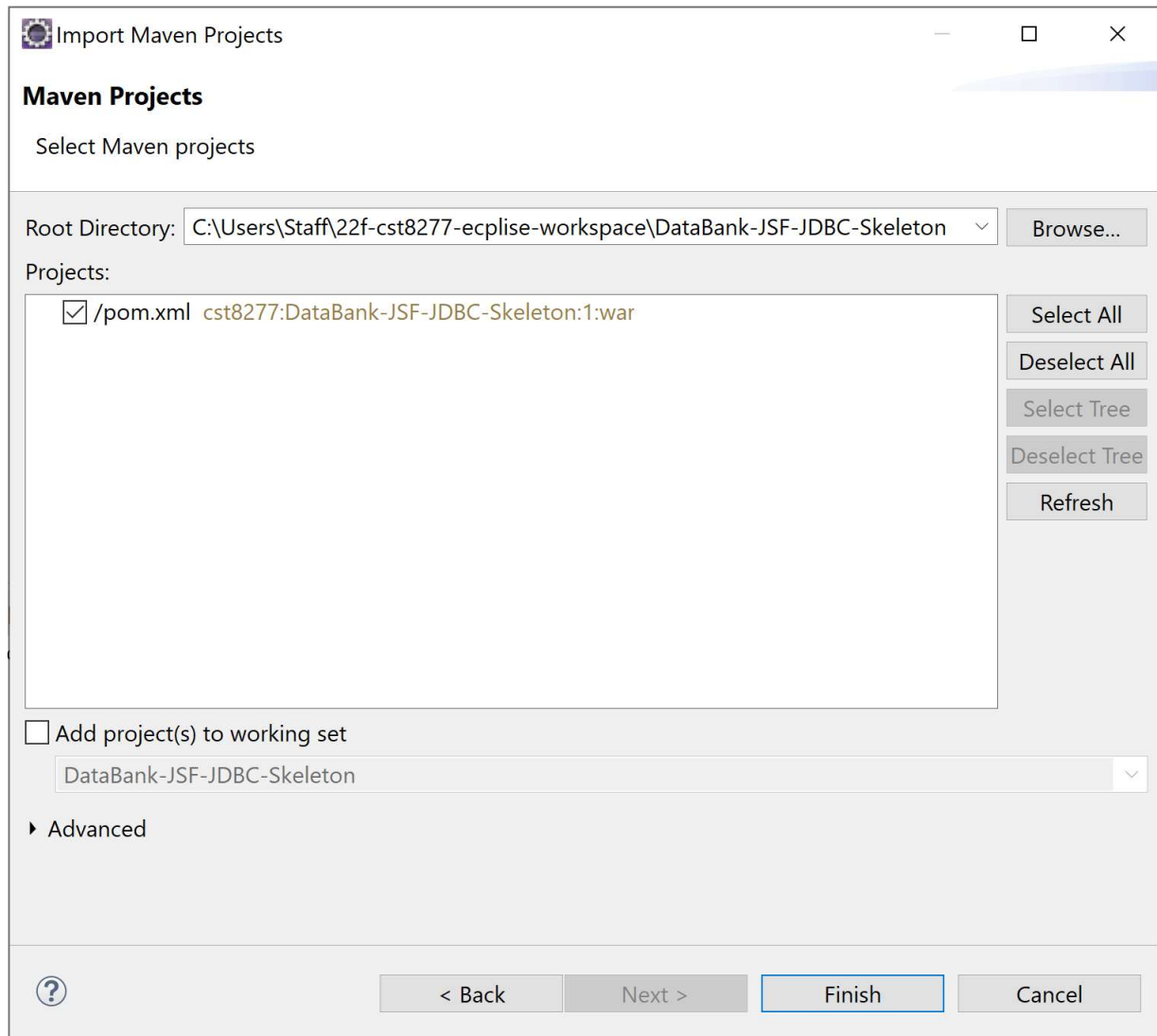
Click on the "Browse…" button above and navigate to the location of the extracted project folder. Make sure that the checkbox beside the pom.xml file above is selected and then click "Finish".

After importing the starter project into your Eclipse workspace, the next step is to download the provided database script **databank1.sql** and execute it on your MySQL Workbench. The database script contains the DDL (data definition language) statements to create the necessary tables in the **databank** database using MySQL syntax:

```sql
CREATE TABLE IF NOT EXISTS `databank`.`student`(
    `id` INT NOT NULL AUTO_INCREMENT,
    `last_name` VARCHAR(50) NOT NULL,
    `first_name` VARCHAR(50) NOT NULL,
    `email` VARCHAR(100) NULL,
    `phone` VARCHAR(10) NULL,
    `level` VARCHAR(10) NULL,
    `program` VARCHAR(60) NULL,
    `created` DATETIME NULL,
    PRIMARY KEY (`id`))
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `databank`.`program`(
    `name` VARCHAR(60) NULL)
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `databank`.`level`(
    `name` VARCHAR(10) NULL)
ENGINE = InnoDB;
```

Running Assignment 1

You may notice that if you try to run the starter project un-changed, it does not show any students. Instead, you will see an internal server error on the main page:

Note that this is _done on purpose_. You must find/investigate and fix the problems as well as extend the project to provide the required CRUD capabilities. To complete the project, you have to locate and complete all the **TODO**s throughout the project as well as understand the existing code and comments provided to you.

To run your project on the Payara server:

- DataBank-JSF-JDBC-Sh...
  - Deployment Descri
  - JAX-WS Web Servic
  - Maven Dependenci
  - src/main/java
  - src/main/resources
  - JRE System Library
  - src/test/java
  - Deployed Resource
  - src
  - target
  - pom.xml
- Payara Server 5 (5.202

New		>
Go Into		
Show In	Alt+Shift+W	>
Copy	Ctrl+C	
Copy Qualified Name		
Paste	Ctrl+V	
Delete	Delete	
Build Path		>
Refactor	Alt+Shift+T	>
Import		>
Export		>
Refresh	F5	
Close Project		
Close Unrelated Project		
Run As		>
Debug As		>
Profile As		>
Restore from Local History...		
Java EE Tools		>
Maven		>
Team		>
Compare With		>
Configure		>
Source		>
Validate		
Properties	Alt+Enter	

1 Run on Server	Alt+Shift+X, R	
2 Java Application	Alt+Shift+X, J	
m2 3 Maven build	Alt+Shift+X, M	
m2 4 Maven build...		
m2 5 Maven clean		
m2 6 Maven generate-sources		
m2 7 Maven install		
m2 8 Maven test		
Run Configurations...		

Data Access Object (DAO) Pattern

In Java EE, there is a "pattern" for Data Access Object (DAO) classes. The following reference is a great introduction: http://ramj2ee.blogspot.in/2013/08/data-access-object-design-pattern- or.html

To connect the controller class with the DAO class, we can **inject** a reference into the controller class' member fields:

```
@Inject
private StudentDao studentDao;

@Inject
private ListDataDao listDataDao;
```

Note: For the DAO, we inject the interface and **not** the implementation!

JSF Tips N' Tricks

In the main JSF presentation view XHTML file **list-students.xhtml**, you must update the <h:viewAction> widget with the appropriate information so that when the application is about to *update its model classes* (Hint: JSF phases), all the students are loaded from the database.

The controller class should be scoped to the HTTP session so that there is only one instance for **http://localhost:8080/databank/list-students.xhtml** (and for navigating back-and-forth with the add/edit pages). If another web browser (or 2nd tab of the same browser) connects to the application, a second controller will be instantiated. **Question: What is the appropriate scope for the DAOImpl class? How about for the model class?**

The presentation view XHTML file responsible for editing an existing model object selected from the list needs a way to communicate between pages. The JSF controller maintains a "session map" that can hold (almost) any object. **Question: How is the sessionMap injected?**

A <h:commandButton> button's action attribute can invoke a method on the controller class which can then put something "special" on the session map (please see code snippets below):

list-students.xhtml:

```
<h:commandButton
     action="#{studentController.doSomething(stud.id)}">
</h:commandButton>
```

StudentController.java:

```
public String doSomething(int id) {
    StudentPojo s1 = studentDao.readStudentById(id);
    sessionMap.put("studentToEdit", s1);
    return "edit-student.xhtml?faces-redirect-true";
}
```

edit-student.xhtml:

```
<h:outputText value="#{uiconsts['columnLabel_FirstName']}"/>
<h:inputText value="#{studentToEdit.firstName}" id="firstName"/>
```

This way, when the edit XHTML form is presented, the existing values are populated into the UI fields.

The add XHTML form requires the same sort thing except that we do not wish to populate the add form with the information from the last selected entity. In the case of add, we put an entirely new empty student model object into the session map and use that on the add page:

```
sessionMap.put("newStudent", new StudentPojo());
```

Submitting Assignment 1

To submit your Assignment 1, first you must export your completed project as a zip file. Eclipse has an export function to create an external archive, i.e. a zip file, of your project. Make sure to not include the "target" subfolder when generating the zip file (see screenshots below). You should name your zip file as **<lastname>-<firstname>-Assignment1.zip**, for example, **Smith-John-Assignment1.zip**. You must then upload the correct generated zip file to Brightspace under Assignment 1.

Please make sure that your completed project compiles and builds successfully, has good indentation and coding style, not overly complicated, and is efficient. The completed code must demonstrate your understanding of how a JSF application works.

The screenshots below show how to export your completed Eclipse project as a zip file:
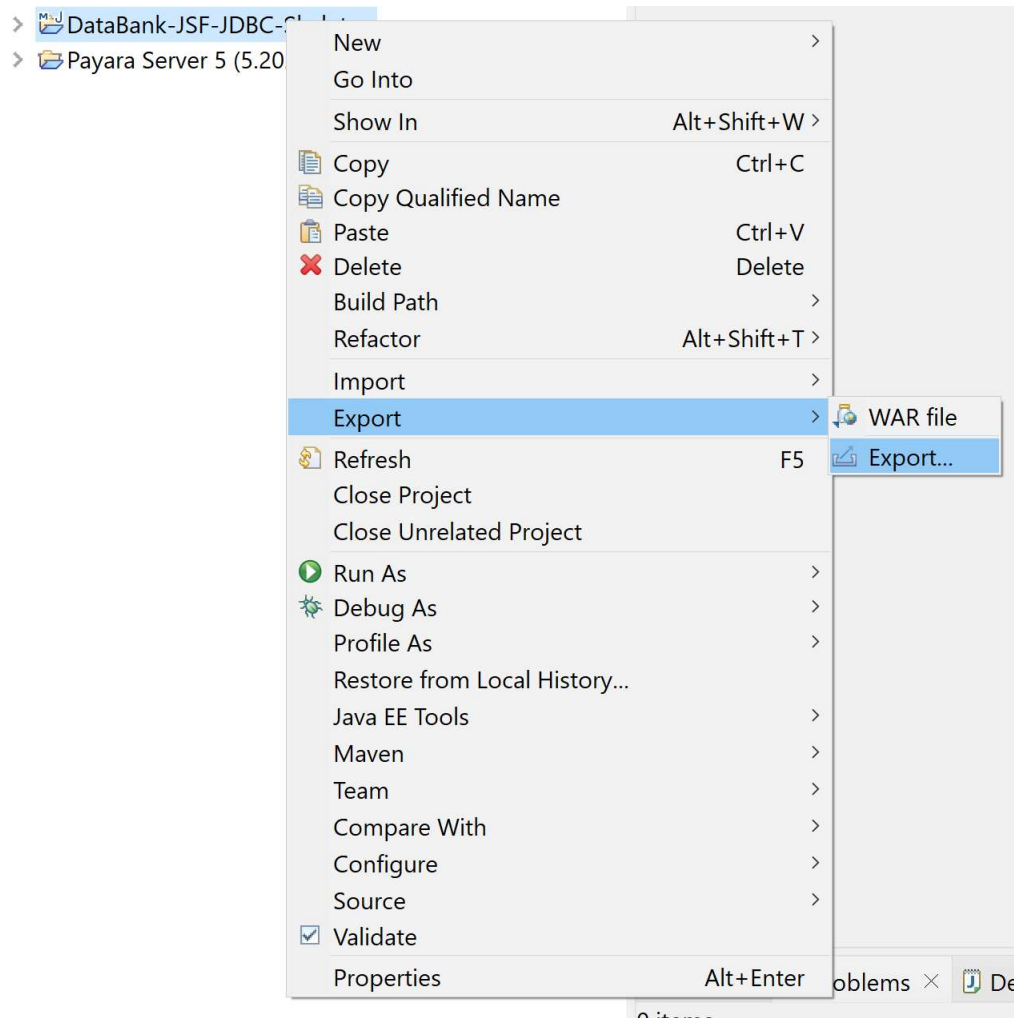
Export

Select

Export resources to an archive file on the local file system.

Select an export wizard:

```
type filter text
```

- ∨ 📂 General
    - 🐜 Ant Buildfiles
    - 🗃 Archive File
    - 📂 File System
    - 🗔 Preferences
- › 📂 EJB
- › 📂 Install
- › 📂 Java
- › 📂 Java EE
- › 📂 Mission Control
- › 📂 Plug-in Development
- › 📂 Run/Debug
- › 📂 Team
- › 📂 Web
- › 📂 Web Services
- › 📂 XML

❓   < Back   Next >   Finish   Cancel

Export

Archive file

Export resources to an archive file on the local file system.

```
☑ ■ DataBank-JSF-JDBC-Skeleton          ☑ x .classpath
  > ☑ .metadata                          ☑ x .project
    ☑ .settings                          ☑ M pom.xml
  > ☑ src
  > ☐ target
> ☐ Payara Server 5 (5.2021.1) [domain1]
```

[Filter Types...] [Select All] [Deselect All]

To archive file: `c:\somefolder\Smith-John-Assignment1.zip`   [Browse...]

Options

◉ Save in zip format   ◉ Create directory structure for files
◯ Save in tar format   ◯ Create only selected directories
☑ Compress the contents of the file
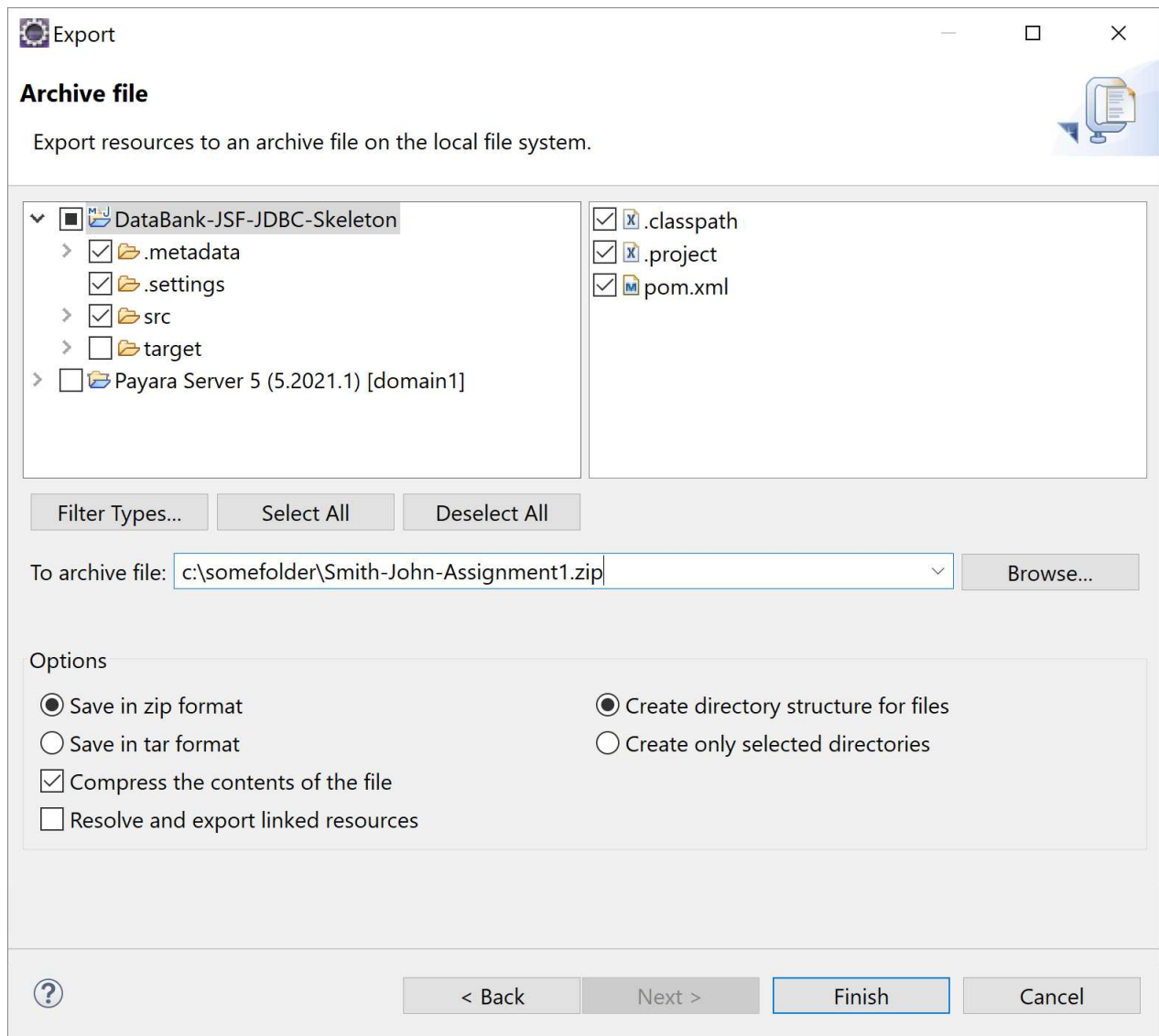☐ Resolve and export linked resources

[< Back] [Next >] [Finish] [Cancel]

Note: Do not forget to upload the correct generated zip file to Brightspace under Assignment 1.

Demoing Assignment 1

For Assignment 1, please note that demo is <u>not mandatory</u> but <u>optional</u>. However, if you think you have completed your project early (before the due date), you can demo your Assignment 1 to your lab professor during your respective lab session before the due date. The advantage of doing an optional demo for your Assignment 1 is that if the demo is done before the due date, then you will have a chance to fix issues/bugs found during the demo before you submit your completed project on Brightspace. Please note that you are allowed to keep submitting updated versions of your project on Brightspace (before the due date) though only the latest version will be considered your official submission and it will be the one to be graded by your lab professor.

– end –